PATENT
Attorney Docket No.: 100111221-1
Express Mail Label No.: EV 210654705 US

SYSTEM AND METHOD FOR DETERMINING WIRE CAPACITANCE FOR A VLSI CIRCUIT

RELATED APPLICATIONS

[0001] The present document contains material related to the material of copending, cofiled, U.S. patent applications Attorney Docket Number 100111227-1, entitled System And Method For Determining Applicable Configuration Information For Use In Analysis Of A Computer Aided Design; Attorney Docket Number 100111228-1, entitled Systems And Methods Utilizing Fast Analysis Information During Detailed Analysis Of A Circuit Design; Attorney Docket Number 100111230-1, entitled Systems And Methods For Determining Activity Factors Of A Circuit Design; Attorney Docket Number 100111232-1, entitled System And Method For Determining A Highest Level Signal Name In A Hierarchical VLSI Design; Attorney Docket Number 100111233-1, entitled System And Method For Determining Connectivity Of Nets In A Hierarchical Circuit Design; Attorney Docket Number 100111234-1, entitled System And Method Analyzing Design Elements In Computer Aided Design Tools; Attorney Docket Number 100111235-1, entitled System And Method For Determining Unmatched Design Elements In A Computer-Automated Design; Attorney Docket Number 100111236-1, entitled Computer Aided Design Systems And Methods With Reduced Memory Utilization; Attorney Docket Number 100111238-1, entitled System And Method For Iteratively Traversing A Hierarchical Circuit Design; Attorney Docket Number 100111257-1, entitled Systems And Methods For Establishing Data Model Consistency Of Computer Aided Design Tools; Attorney Docket Number 100111259-1, entitled Systems And Methods For Identifying Data Sources Associated With A Circuit Design; and Attorney Docket Number 100111260-1, entitled Systems And Methods For Performing Circuit Analysis On A Circuit Design, the disclosures of which are hereby incorporated herein by reference.

BACKGROUND

[0002] Computer aided design (CAD) tools, including computer aided engineering (CAE) tools, are used to create and analyze various types of designs. An E-CAD tool is a type of CAD tool that is used for creating and analyzing electronic designs. E-CAD tools that analyze VLSI designs typically operate on a single type of wire capacitance data. It is advantageous, however, to allow analysis tools to operate on the best available data, which may originate from different data sources, so that analysis may be performed on blocks of a VLSI design that are not completely drawn or laid out, for example. An existing technique attempts to generate a hierarchical data model from mixed-mode data sources, but this technique may generate inconsistencies in the data model that result from mismatched connectivity data, thereby preventing accurate circuit analysis.

SUMMARY

[0003] A process of determining wire capacitance for a VLSI circuit design is described. In one embodiment, the process comprises determining all hierarchical blocks of a portion of the design; storing, for a plurality of the blocks, indicia of the most accurate one of a plurality of wire capacitance data sources; generating a wire capacitance database with an entry for each net, in at least a plurality of the blocks, using information stored in at least one of the wire capacitance data sources; generating a hierarchical connectivity model for the design; and using the hierarchical connectivity model and the wire capacitance database to determine a cumulative wire capacitance value for each HLSN (highest level signal name) in each of the blocks in the portion of the design to be analyzed.

[0004] In one embodiment, the hierarchical connectivity model for the design is generated by using layout connectivity, if extracted layout data is available for a given block; otherwise, if no layout is available for the block, then by using schematic connectivity data.

[0005] In an alternative embodiment, the hierarchical connectivity model for the design is generated by using a single type of connectivity data for each of the blocks, where a single type of data is selected from either layout or a schematic diagram.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] Figure 1 is a block diagram of one embodiment of an exemplary CAD system for selecting wire capacitance data for a VLSI netlist;

[0007] Figure 2 is a flowchart illustrating exemplary steps that may be performed during operation of the CAD system of Figure 1; and

[0008] Figure 3 is an example of a portion of a hierarchical design to be analyzed by the CAD system of Figure 1.

DETAILED DESCRIPTION

DEFINITIONS

[0009] A net is a single electrical path in a circuit design that has the same electrical characteristics at all of its points. Any collection of wires that carries the same signal between components is a net. If the components allow the signal to pass through unaltered (as in the case of a terminal), then the net continues on subsequently connected wires. If, however, the component modifies the signal (as in the case of a transistor or a logic gate), then the net terminates at that component and a new net begins on the other side. In the present document, a net may be considered to be subdivided into net 'pieces', each of which is part of a 'highest level signal name', or HLSN. An HLSN is the unique signal name that identifies a collection of local block nets or 'hierarchical net pieces', which are the small pieces of wire (nets) in each hierarchical block of a circuit design. A netlist lists the components and the nets associated with the circuit design.

[0010] A significant characteristic of VLSI and other types of circuit design is a reliance on hierarchical description. A primary reason for using hierarchical description is to hide the vast amount of detail in a design. By reducing the distracting detail to a single object that is lower in the hierarchy, one can greatly simplify many CAD operations. For example, simulation, verification, design-rule checking, and layout constraints can all benefit from hierarchical representation, which makes them more computationally tractable. Since many circuits are too complicated to be easily considered in their totality, a complete design is often viewed as a collection of component aggregates that are further divided into sub-aggregates in a recursive and hierarchical manner. In VLSI design, these aggregates are commonly referred to as blocks (or cells); the use of a block at a given level of hierarchy is called an 'instance'.

[0011] Figure 1 is a block diagram of one embodiment of an exemplary CAD system 100 configured for selecting wire capacitance data for a VLSI design 109, in accordance with one embodiment of the present system. As shown in Figure 1, CAD system 100 includes computer system 101 and CAD tool 107. Design 109 includes a netlist 105 for a hierarchical VLSI design developed by CAD tool 107. Computer system 101 includes processor 102, computer memory 104, and storage unit 106. In computer system 101, processor 102 is coupled to computer memory 104 and to storage unit 106.

[0012] In one embodiment of the present system, CAD tool 107 may initially reside in storage unit 106 as software instructions. Upon initializing CAD tool 107, software instructions that form CAD tool 107 are loaded in computer memory 104. At least part of design 109, including netlist 105 is loaded in computer memory 104 upon initialization of CAD tool 107. Processor 102 then executes the software instructions, as described below with respect to Figure 2.

[0013] In CAD system 100, processor 102 controls operation of the system, and also functions to combine data from wire capacitance value sources 110(1), 110(2) ... 110(N), hereinafter referred to as wire capacitance sources 110(*). Wire capacitance sources 110(*) may include user input, design analyzers, CAD tool estimators, and the like. In CAD system 100, processor 102 is coupled to each of wire capacitance sources 110(*) through link 111. For example, source 110(1) may be a user input that provides a wire capacitance value of a first design element, while source 110(2) may be a CAD tool estimator that provides a wire capacitance value of a second design element, and source 110(N) may provide a value of wire capacitance for a third design element of the design as determined by the analysis of the design by an E-CAD tool. The term 'design element' may be used to refer to any component that is part of a design, and, as used herein, typically refers to a net, which is a wire or other element, that connects two components of a VLSI circuit. It is to be noted that the present system may determine design element characteristics other than wire capacitance.

[0014] The wire capacitance values used by the present system may, for example, be generated from an analysis of a design block by a wire capacitance source 110(*) such as a design analyzer and/or a CAD tool estimator. Examples of types of analyses performed by such a source 110(*) includes, for example, analyzing capacitances, resistances, and leakage currents of various design elements. Each of wire capacitance sources 110(*) generates a value for the wire capacitance of a particular net and a data source indicator 103 that indicates the source of the wire capacitance data (e.g., wire capacitance values acquired through an E-CAD tool functioning as a capacitance estimator source).

[0015] In an exemplary embodiment of the present system, processor 102 uses the 'best available' wire capacitance data, i.e., data from the source considered by a design engineer to be the most accurate, for each of the nets of a hierarchical VLSI design. For each block within the hierarchical design, the best available data source [e.g., one of sources 110(*)] for analyzing the block is stored by name in data source database 108 of storage unit 106. Processor 102 reads entries in a wire capacitance sub-database 112(*), based on the block name in data source database 108, to generate wire capacitance values for each block under analysis based on each data source in database 108. Processor 102 additionally functions as a summation tool that reads each value associated with each source and sums the values for the associated block of the design. Processor 102 may also generate an estimation of the particular characteristic if any values stored in data source database 108 contain missing information.

[0016] It should be noted that processor 102 may incorporate the functionality of sources 110(*). For example, items 110(1), 110(2) ... 110(N) may take the form of one or more software modules resident within computer memory 104 and/or inputs to computer memory 104.

[0017] Figure 2 is a flowchart illustrating exemplary steps performed during operation of the CAD system 100 of Figure 1. Figure 3 is an example of a portion of a hierarchical design to be analyzed by the CAD system 100 of Figure 1, and represents a hierarchical connectivity model, as explained below with respect to step 215. Operation of the present system is best understood by viewing Figures 1, 2, and 3 in conjunction with one another. As shown in Figure 2, operation 200 commences at step 202, wherein processor 102 determines all of the hierarchical blocks for the portion of design to be analyzed. In the following example, the design portion 300 to be analyzed is shown in Figure 3 as consisting of a highest level block 'inv-pair', which comprises two hierarchically lower blocks 'inv1' and 'inv2', each of which is an instance of block 'inv*'.

[0018] At step 205, for each hierarchical block in the design portion of interest, processor 102 retrieves the best available wire capacitance data source name from a configuration file in database 113, and stores this name and the type of data source (e.g., layout or schematic connectivity) in data source database 108 in a record associated with the corresponding block. There is only one data source database 108 for the entire run of the present analysis, and that database has one record for each block in the design portion of interest. In the present example, assume that the 'best available' source name for block 'inv-pair' is layout 'source_110(1)', and the 'best available' data source for both blocks 'inv1' and 'inv2' is schematic 'source_110(2)', since both of these blocks are instances of the same block, 'inv*'. In an exemplary embodiment, each record created in data source database 108 includes the block name, the type of data source, and the name of the data source directory and data file therein containing the wire capacitance data for the associated block, as shown in Table 1 below (in which other database entries and other information have been omitted for simplicity).

TABLE 1: DATA SOURCE DATABASE 108

Block Name	Type	Data Source Directory/Data File
inv-pair	layout	source_110(1)/hier_cap.1
inv*	schematic	source_110(2)/ hier_cap.2

[0019] In the present example, it can be seen from Table 1 that the 'best available' data for the 'inv-pair' block is layout data, which is found in the data source directory 'source_110(1)'; and that the corresponding wire capacitance for each net in that block can be found in wire capacitance sub-database 'hier_cap.1'. Likewise, the 'best available' data for the 'inv*' block is schematic data, which is found in the data source directory 'source_110(2)'; and the corresponding wire capacitance for each net in that block can be found in wire capacitance sub-database 'hier_cap.2'.

[0020] At step 210, a plurality of wire capacitance sub-databases 112(*) are generated by processor 102, by retrieving wire capacitance data from one or more data sources 110(*) to produce an entry in the wire capacitance database for each block net in the design portion of interest. Each wire capacitance sub-database 112(*) contains a record, for each block in a particular net, that provides the wire capacitance for that net. There is a wire capacitance sub-database 112(*) for each block in the portion of the design to be analyzed. Since all nets in a block share the same data source, it is not necessary to associate a data source with each net, but only to maintain the data source – block name association, as shown in Table 1.

[0021] Records for block nets inv-pair/conn, inv*/in, and inv*/out are shown in Table 2, below (other block nets in the design portion of interest are omitted for simplicity):

TABLE 2: WIRE CAPACITANCE SUB-DATABASES 112(*)

Data source directory: source_110(1)/

Wire capacitance sub-database: hier_cap.1:

Net	Capacitance
conn	0.03 pF

[Other sub-databases may also exist]

Data source directory: source_110(2)/

Wire capacitance sub-database: hier_cap.2:

Net	Capacitance
in	0.01 pF
out	0.02 pF

[0022] The location of the wire capacitance data for a particular net may be determined by locating the record for the appropriate block (i.e., the block in which the net is located) in the data source database 108, which record contains the directory name for the data source directory containing the wire capacitance sub-database of interest. It can be seen from Table 2, that 'source_110(1)' provides a capacitance 0.03 pF for the net 'conn' in the block 'inv-pair', and that 'source_110(2)' provides respective capacitances of 0.01 pF and 0.02 pF for the nets 'in' and 'out' associated with the block 'inv*'.

[0023] At step 215, a hierarchical connectivity model 300 is generated for the design portion of interest. In an exemplary embodiment, the hierarchical connectivity model 300 is generated using a single type of data for each block. In this step, processor 102 reads in connectivity data (typically either schematic or layout connectivity) from data of design 109 that provides the full hierarchy of the design portion to be analyzed.

[0024] In the exemplary embodiment described above, the present system may generate a mixed-mode result (i.e., including wire capacitance values originating from different data sources) without reading in mixed-mode connectivity information, which often does not produce accurate results. In some cases, the layout hierarchy and schematic hierarchy do not match, and, therefore, when layout connectivity is used for some blocks and schematic connectivity for others, a mismatched hierarchy results which prevents accurate circuit analysis. This embodiment avoids this problem of mismatched hierarchy by selecting either layout or schematic connectivity, and then loading the proper capacitance data onto the connectivity model. Either layout or schematic connectivity is chosen based on what the data source database108 contains; if all sources for all blocks in the circuit hierarchy are layout related, then the layout connectivity is used. If, however, any of the blocks use schematic or some other connectivity representation (estimated parasitics from another tool, for example), then the schematic connectivity information is used to construct the hierarchical connectivity model. In the present example, assume that the block 'inv-pair' is generated from schematic data, and that each of the blocks 'inv1' and 'inv2' are generated from an instance of 'inv*' which is provided by layout data. If a block has layout available, schematic data is always available; therefore, schematic connectivity information is used to construct the hierarchical connectivity model, as shown in Figure 3.

[0025] In an alternative embodiment, the hierarchical connectivity model is generated by using whichever type of connectivity data is appropriate for each particular block. That is, if layout data is available for a given block, layout connectivity is used, and if no layout is available for a block, then schematic connectivity data is used.

[0026] At step 218, the hierarchical connectivity model 300 generated in step 215 and data in the wire capacitance sub-database 112(*) is used to sum wire capacitance for each HLSN in the design portion to be analyzed, while maintaining a data source indicator 103(*) associated with each HLSN (where the '*' character indicates a specific HLSN). In an exemplary embodiment, data source indicators 103(*) are bit 'vectors' in which each bit in the vector indicates a specific data source, although any other type of mechanism for associating an HLSN wire capacitance value with a data source may be used.

[0027] In the present exemplary embodiment, an empty data source indicator 103(*) is initially created for each HLSN to be analyzed. As indicated above, all nets in a block have the same source, and there is one sub-database 112(*) per block, as shown in Table 2. The data source database 108 is read and its contents are kept in memory as the connected nets on each HLSN are traversed to determine the wire capacitance for a given HLSN. During this process, the block name in which a particular net piece resides is looked up in source database 108 to get the source type of the wire capacitance data, and the wire capacitance database 112(*) for a particular net is also referenced to determine the wire capacitance value for that net.

[0028] In the present example, the HLSN 'conn', which includes the component nets (i.e., net pieces) 'out', 'conn', and 'in', is traversed to determine the wire capacitance for the HLSN 'conn'. The data source directory for the net 'conn' is determined by referring to block 'inv-pair' in data source database 108, since 'conn' is located in the block 'inv-pair'. In similar fashion, the data source directory for the nets 'in' and 'out' is determined by referring to block 'inv-pair' in source database 108, since both nets 'in' and 'out' are located in the block 'inv*'. If there is no corresponding data source found in data source database 108 for any component or element of a particular HLSN, then a 'missing data' bit, indicating that a wire capacitance value is missing, is set in the associated data source indicator 103(*). In certain situations, wire capacitance data may be missing from the wire capacitance database 112(*) for a particular net. Thus, there are two types of potentially missing data: a missing entry in the source database 108; and a missing entry in a particular wire capacitance database 112(*). These two types of missing data can be represented by two different bits being set in data source indicator 103(*), which in the present embodiment, is a bit vector. An abbreviated example of a bit vector format (a typical bit vector may have 32 bits) for data source indicator 103(*) is shown in Table 4 below, with an indication of the significance of indicator bits 0 –3:

TABLE 4: SAMPLE BIT VECTOR 103(*) FORMAT

Bit Number	3	2	1	0
Indication	missing source DB entry	missing wire cap. DB entry	data source 110(1)	data source 110(2)

[0029] In the present example, the data source indicator 103(conn) is generated for the HLSN 'conn', as a bit vector with the following indicia (indicator bits):

– a bit indicating that the data source for the net 'conn' is 'source_110(1)'; and

– a bit indicating that the data source for the nets 'in' and out' (which are both instances of block inv*) is 'source_110(2).

The resulting data source indicator/bit vector103(conn) (abbreviated to show only the four bits in Table 4) thus has the value '0011'. If it were determined, for example, that there was no data in wire capacitance database 112(*) for one of the net pieces for the HLSN 'conn', then the resulting bit vector103(conn) would have the value '0111'.

[0030] In the present example, the wire capacitance is determined for the HLSN 'conn', which represents the local block nets (or hierarchical net pieces) 'out' (in block 'inv1'), 'conn', and 'in' (in block 'inv2'). As previously defined, an HLSN is the unique signal name that identifies a collection of block nets. In the present example, the signal name 'conn' is identical to the block net 'conn' in inv_pair1. If, for example, the block inv_pair1 were instantiated in a parent block, and the instantiation identified as inv_pair_X, the HLSN would be 'inv_pair_X/conn', which indicates that it is a unique signal name in the parent block's hierarchy. Therefore, when analyzing HLSNs, data is accumulated for an HLSN by collecting data on all of the block nets, including the block net that exists at the highest level, and which provides the signal with its name ('conn' in the present case). The wire capacitance for the HLSN 'conn' (which consists of the net pieces 'in/conn/out') is thus determined from wire capacitance sub-databases 112(hier_cap.1) and 112(hier_cap.2) by summing the wire capacitance values therein for net pieces 'out' (0.02 pF), 'conn' (0.03 pF), and 'in' (0.01 pF), to yield a cumulative wire capacitance value of 0.06 pF. This value is stored in a cumulative wire capacitance indicator 114(conn) for the HLSN 'conn'. At this point, the data source indicator [103(conn)] for the HLSN 'conn' now contains two 'set' bits indicating that the cumulative wire capacitance value contained in indicator 114(conn) was determined from two sources, specifically 'source_110(1)' and 'source_110(2)'.

[0031] At step 220, processor 102 determines whether there are any missing wire capacitance values by checking the 'missing data' bit in the corresponding data source indicator 103(*) to determine if there is a missing wire capacitance value for any particular net piece. If it is determined that one or more values are missing, then at step 222, any missing values are generated by processor 102, using any desired method. For example, a minimum wire capacitance value may be used for all missing data, or missing data may be generated from design data relating to total signal length and the number of connected devices. The estimation of missing net pieces may be necessitated by mismatches in the hierarchy between layout connectivity and schematic connectivity. For example, if the connectivity representation that is chosen is schematic-based (since not all hierarchical blocks had layout-based data as their best available source), it is possible that a wire capacitance DB may not exist for a block that appears in the schematic hierarchy, because the block name is different, or because that block does not appear in the layout connectivity. In addition, a wire capacitance estimation may be required in the event that a wire capacitance value is nonexistent in source database 108, or if there is a missing entry in a particular wire capacitance database 112(*). Analysis then proceeds with step 230, described below.

[0032] If it is determined in step 220 that there are no missing wire capacitance values, then at step 225 a check is made to determine whether all of the wire capacitance data for a particular block is from the most accurate, or 'highest quality source'. Note that the data in the wire capacitance databases 112(*) is the 'best available' data for each block. Using this data, however, may not result in a wire capacitance sum that is entirely composed of 'highest quality' data. Ranked from highest quality to lowest quality, the available wire capacitance data may be in the form of layout, schematics, or estimated with a tool that provides an estimate of the capacitance of a wire, given minimal design information. If the wire capacitance sum is generated entirely from 'highest quality' sources, i.e., from layout data, then system operation proceeds at step 235, wherein, the sum generated in step 218 is used in the circuit analysis. If, however, the wire capacitance sum is not generated entirely from 'highest quality' sources, a wire capacitance estimation is performed (as described above with respect to step 222). To make this determination, the data source indicator 103(*) is examined to determine which sources were used to generate the total wire capacitance value calculated in step 218. The estimate that is generated is based on the design data in memory, but the estimate does not rely on the wire capacitance from the wire capacitance databases 112(*); rather it is a separate number that is used to provide a check against the (possibly optimistic) wire capacitance databases. The higher of the two values calculated in steps 218 and step 222 is then used in the circuit analysis, at step 230.

[0033] In the present example, the data source indicator 103(conn) for the HLSN 'conn' contains two 'set' bits indicating that the cumulative wire capacitance value contained in wire capacitance indicator 114(conn) was determined from two sources, specifically 'source_110(1)' and 'source_110(2)'. The data source database 108 is then checked to determine that the 'best available' source name for block 'inv-pair1' (which contains the net piece 'conn') is 'source_110(1)', which is layout, and the 'best available' data source for block inv* (which contains net pieces 'out' and 'in') is 'source_110(2)', which is schematic data. Therefore, since the wire capacitance data used to generate the sum stored in wire capacitance indicator 114(conn) was not generated entirely from 'highest quality' sources, a wire capacitance estimation is performed, and the higher of the two values calculated in steps 218 and step 222 is used in the analysis of design 109.

[0034] Instructions that perform the operation discussed with respect to Figure 2 may be stored on computer-readable storage media. These instructions may be retrieved and executed by a processor, such as processor 102 of Figure 1, to direct the processor to operate in accordance with the present system. The instructions may also be stored in firmware. Examples of storage media include memory devices, tapes, disks, integrated circuits, and servers.

[0035] Certain changes may be made in the above methods and systems without departing from the scope of the present system. It is to be noted that all matter contained in the above description or shown in the accompanying drawings is to be interpreted as illustrative and not in a limiting sense. For example, the items shown in Figure 1 may be constructed, connected, arranged, and/or combined in other configurations, and the set of steps illustrated in Figure 2 may be performed in a different order than shown without departing from the spirit of the present invention.